# HTML & CSS essentials

## 01 HTML tags

### ---CONFIGURATION ELEMENTS

HTML

<html lang=”en”>

Meta

<meta charset=”UTF-8” />

<meta name="description" content="Omnifood is an AI-powered food subscription that will make you eat healthy again, 365 days per year. It's tailored to your personal tastes and nutritional needs." />

Title

<title></title>

Link

* Favicon: on tab browser (64x64px)

<link rel="icon" href="img/favicon - Copy.png" />

* Favicon: Apple favorite (180x180px)

<link rel="apple-touch-icon"

      href="img/apple-touch-icon.png" />

* Favicon: Android favorite (192x192px & 512x512px) – create a new file called *manifest.webmanifest*:

{

  "icons": [

    { "src": "img/favicon 192.png", "type": "image/png", "sizes": "192x192" },

    { "src": "img/favicon 512.png", "type": "image/png", "sizes": "512x512" }

  ]

}

Then in HTML:

<link rel="manifest" href="manifest.webmanifest" />

* To link CSS file to HTML file:

 <link rel="stylesheet" href="css/style.css" />

Script - consider using *defer* or *async* if inserted inside *head* element (refer to JavaScript notes)

<script src=”script.js”></script>

### ---TEXT ELEMENTS

Headings

<h1></h1>

<h2></h2>

<h3></h3>

…

<h6></h6>

Normal paragraphs

<p></p>

Bold text

<b></b>

Italic text

<i></i>

Strong text (semantic)

<strong></strong>

Emphasized text (semantic)

<em></em>

Ordered lists

<ol>

<li></li>

<li></li>

</ol>

Unordered lists

<ul>

<li></li>

<li></li>

</ul>

### ---IMAGE ELEMENTS

<img/>

#### +ATTRIBUTES

Src (link to a directory or a URL) and alt (for screen readers)

<img src=”image.jpg” alt=”html” />

Width and height

<img src=”image.jpg” alt=”html” width=”500” height=”200”/>

#### Tips

* When placing an image within an empty div element, we should use role and aria-label attributes to make screen readers able to read the image.

<div class=”cta-imgbox” role=”img” aria-label=” description of the background image”></div>

### ---HYPERLINKS

<a></a>

#### +ATTRIBUTES

Target (to open link in a new page, set to *\_blank*)

<a href="mdn.html" target="\_blank">MDN Web Docs</a>.

Href (to another page, to another website, to call, to Email, to another section with *id* attribute)

<a href="https://www.mdn.com/">MDN Web Docs</a>

<a href=”tel:05138467125”>call</a>

<a href=”mailto:omidarmat@gmail.com”>email</a>

* To scroll to a certain section that has an *id* attribute:

**HTML:**

Section

<section class=”section-cta” id=”cta”></section>

Hyperlink:

<a href=”#cta” class=”btn”>Start eating well</a>

**CSS** (will probably not work in Safari)**:**

html {

scroll-behavior: smooth;

}

### ---NAVIGATIONS

Nav (Semantic – usually for navigation within the website)

<nav></nav>

Menu (usually for menu buttons for a web application)

<menu></menu>

### ---SECTIONS

Div

<div></div>

Main

<main></main>

Article

<article></article>

Header

<header></header>

Footer

<footer></footer>

Aside

<aside></aside>

### ---TABLES

General table tag structure

<table>

<tr>

<td></td>

</tr>

</table>

Table with head and body

<table>

<thead>

<th></th>

</thead>

<tbody>

<tr></tr>

<t/body>

#### CSS properties:

Table borders

table {

Border: 1px solid #fff;

Border-collapse: collapse;

}

Cell borders

tr, th {

Border: 1px solid #fff;

}

Cell dimensions

thead th {

Width: 25%;

}

### ---BUTTONS

Button

<button></button>

Link

<a></a>

### ---FORMS

Form container tag

<form></form>

Select list (+ options)

<select>

<option value=”friends”>Friends</option>

<option value=” family”>Family</option>

</select>

* To make the selection a necessary task, put *required* in the select opening tag.

<select required></select>

Label tags (use *for* attribute in connection with *id* attribute of input tag)

<label for=”<input-id>”></label>

Input tags (use *id* attribute in connection with *for* attribute of label tag)

<input />

#### Input tags attributes:

Type (type of input: text, email, submit, number, checkbox, etc.)

<input type=”text”/>

* For textbox inputs, *value* is what user types.

Placeholder (hint)

<input type=”text” placeholder=”John Smith” />

* To make the selection a necessary task, put *required* in the select opening tag.

<input type=”text” required></input>

## 02 HTML entities

### ---Copyright sign

&copy;

Check Jonas resources page for more entities.

## 03 CSS selectors

#### Selector priority:

1. Declarations marked as **!important**
2. **Inline style** (style attribute in HTML)

We don’t use this and it does not have much importance.

1. **ID selector** (#)

If there are multiple ID selectors, the last one applies. It is a bit complicated than that and it will be discussed later.

1. **Class** (.) or **pseudo-class** (:) selector

If there are multiple class or pseudo-class selectors, the last one applies.

1. **Element selector** (p, div, li, etc.)

Multiple? The last one applies.

1. **Universal selector** (\*)

#### Inheritance:

Not all the properties get inherited. It’s mostly ones related to text that get inherited:

Font-family, font-size, font-weight, font-style, color, line-height, letter-spacing, text-align, text-transform, text-shadow, list-style, etc.

### ---UNIVERSAL SELECTOR

\*{

Margin:0;

Padding:0;

Box-sizing: border-box;

}

### ---ELEMENT SELECTOR

h1{}

### ---LIST SELECTOR

h1, h2, h3, h4, p, li {}

### ---DESCENDANT SELECTOR

Footer p{}

### ---AND SELECTOR

.class1.class2 {}

### ---CLASS SELECTOR

.related-author {}

### ---ID SELECTOR

#copyright {}

### ---PSEUDO-CLASS SELECTOR

li:first-child { }

li:last-child {}

li:nth-child(2){}

li:nth-child(odd){}

li:nth-child(even){}

a:hover {}

a:visited {}

a:active {}

### ---PSEUDO-ELEMENT SELECTOR

h1::first-letter {}

p::first-line{}

h2::after {

Content:”TOP”;

Display: inline-block;

Position: absolute;

}

h2::before {

Content:”TOP”;

Display: inline-block;

Position: absolute;

}

### ---ADJACENT SIBLING SELECTOR

h2 + p::first-line {}

### ---SELECTING WITH ANY ATTRIBUTE

HTML:

<button class=”icon” name=”close-outline”></button>

CSS:

.icon[name=”close-outline”] {}

or

HTML:

<img

src = “img/digital-lazy.jpg”

data-src = “img/digital.jpg”

class = “features\_\_img”

/>

CSS:

img[data-src] {}

### ---CSS CUSTOM PROPERTIES

In the CSS file, in a selection called *root* we can specify some custom properties. It works based on the same concept as variables.

:root {

--color-primary: #5ec576;

--gradiant-secondary: linear-gradient(to top left, #39b385, #9be15d)

}

These custom properties can then be accessed from the JavaScript file.

### ---TYPES OF BOXES

Display: block

Display: inline

Display: inline-block

### ---FLEXBOX

#### Flex container:

* Gap: 0 | <length>

It is used to create space between items, without using margin

* Justify-content: flex-start | flex-end |center

It is used to align items horizontally along main axis (horizontally, by default)

* Align-items: stretch | flex-start | flex-end | center | baseline

It is used to align items vertically along cross axis (vertically, by default)

* Flex-direction: row | row-reverse | column | column-reverse

It is used to define which is the main axis

* Flex-wrap: nowrap | wrap | wrap-reverse

To allow items to wrap into a new line if they are too large

* Align-content: stretch | flex-start | flex-end | center | space-between | space-around

This only applies when there are multiple lines (flex-wrap: wrap)

#### Flex items:

* Align-self: auto | stretch | flex-start | flex-end | center | baseline

It is used to overwrite align-items for individual flex items

* Flex-grow: 0 | <integer>

It is used to allow an element to grow (0 means no, 1+ means yes)

* Flex-shrink: 1 | <integer>

It is used to allow an element to shrink (0 means no, 1+ means yes)

* Flex-basis: auto | <length>

It is used to define an item’s width, instead of the width property

* Flex: 0 1 auto | <int> <int> <length>

Recommended shorthand for flex-grow, flex-shrink, flex-basis

* Order: 0 | <integer>

Controls order of items. -1 makes item first, 1 makes it last.

#### Flexbox tips:

* When we have a div element (or any other block element) with text inside it, we can set the div to display as a flex container.

### ---CSS-GRID

#### Grid container:

* Grid-template-rows: <track size>\*
* Grid-template-columns: <track size>\*

To establish the grid row and column tracks. One length unit for each track. Any unit can be used, new **fr** fills unused space.

* Row-gap: 0 | <length>
* Column-gap: 0 | <length>
* Gap: 0 | <length

To create empty space between tracks.

* Justify-items: stretch | start | center | end
* Align-items: stretch | start | center | end

To align items inside rows / columns (horizontally / vertically)

* Justify-content: start | center | end
* Align-content: start | center | end

To align entire grid inside grid container. Only applies if container is larger than the grid.

#### Grid items:

* Grid-column: <start line> / <end line> | span <number>
* Grid-row: <start line> / <end line> | span <number>

To place a grid item into a specific cell, based on line numbers.

* Justify-self: stretch | start | center | end
* Align-self: stretch | start | center | end

To overwrite justify-items / align-items for single items.

### ---UNITS:

* px
* %
* vh & vw
* fr (fraction: CSS grid dimensions)
* rem (use together with font-size set to 62.5% in html tag)
* em (for media queries)

### ---SOME SPECIAL PROPERTIES:

Background image

background-img: linear-gradient(rgba(0,0,0,0.6), rgba(0,0,0,0.6)), url(hero.jpg)

background-image: linear-gradient (90deg, #333, #579755)

Background size

background-size: cover;

Overflow

overflow: scroll

overflow: hidden

Box shadow (horizontal offset, vertical offset, blur amount, scale the shadow up, color)

box-shadow: 20px 20px 20px 10px #000

Text shadow (same as box shadow, except the 4th parameter)

text-shadow: 0 5px 5px #000;

Border radius (if square: 50%, if not: high px value)

border-radius: 12px;

border-bottom-left-radius: 0;

border-bottom-right-radius: 0;

Transition

transition: all 0.3s

Transform

transform: scale(2);

Transform: translate (0, -50%);

Backdrop (may not work in all browsers. Use -webkit- if necessary)

backdrop-filter: blur(10px)

blur

filter: blur(20px);